Bayesian Networks

C. Learning / C.3 Structure Learning / PC Algorithm

Lars Schmidt-Thieme

Information Systems and Machine Learning Lab (ISMLL)
Institute of Computer Science
University of Hildesheim
http://www.ismll.uni-hildesheim.de

Syllabus

Mon. 24.10.	(1)	0. Introduction
		A. Fundamentals
	&	A.1. Probability Calculus revisited
Mon. 31.10.	(2)	A.2. Separation in Graphs
Mon. 7.11.	(3)	A.3. Bayesian and Markov Networks
Mon. 14.11.	(4)	(ctd.)
		B. Inference
Mon. 21.11.	(5)	B.1. Exact Inference
Mon. 28.11.	(6)	B.2. Exact Inference / Join Tree
Mon. 5.12.	(7)	(ctd.)
Mon. 12.12.	(8)	B.3. Approximate Inference / Sampling
Mon. 19.12.	(9)	B.4. Approximate Inference / Propagation
		C. Learning
Mon. 9.1.	(10)	C.1. Parameter Learning
Mon. 16.1.	(11)	C.2. Parameter Learning with Missing Values
Mon. 23.1.	(12)	C.3. Structure Learning / PC Algorithm
Mon. 30.1.	(13)	C.4. Structure Learning / Search Methods

1. PC Algorithm

Types of Methods for Structure Learning

There are three types of structure learning algorithms for Bayesian networks:

1. **constrained-based learning** (e.g., PC),
2. **searching with a target function** (e.g., K2),
3. **hybrid methods** (e.g., sparse candidate).

Computing the Skeleton

Lemma 1 (Edge Criterion)**.** *Let* $G := (V, E)$ *be a DAG and* $X, Y \in V$*. Then it is equivalent:*

(i) X and Y cannot be separated by any $\mathcal{Z}$, i.e.,

$$\neg I_G(X, Y \mid \mathcal{Z}) \quad \forall \mathcal{Z} \subseteq V \setminus \{X, Y\}$$

(ii) There is an edge between X and Y, i.e.,

$$(X, Y) \in E \text{ or } (Y, X) \in E$$

Definition 1. Any $\mathcal{Z} \subseteq V \setminus \{X, Y\}$ with $I_G(X, Y \mid \mathcal{Z})$ is called a **separator of** X **and** Y.

$$\mathrm{Sep}(X, Y) := \{\mathcal{Z} \subseteq V \setminus \{X, Y\} \mid I_G(X, Y \mid \mathcal{Z})\}$$

Computing the Skeleton / Separators

```
separators-basic(set of variables V, independency relation I) :
Allocate S : P^2(V) → P(V) ∪ {none}
for {X, Y} ⊆ V do
    S({X, Y}) := none
    for T ⊆ V \ {X, Y} do
        if I(X, Y | T)
            S({X, Y}) := T
            break
        fi
    od
od
return S
```

Figure 1: Compute a separator for each pair of variables.

Example / 1/3 – Computing the Skeleton

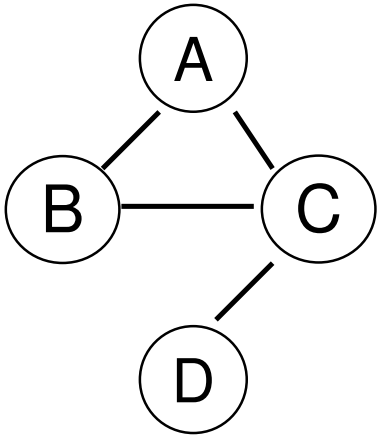
Let I be the following independency structure:

$$I(A, D \,|\, C), \quad I(A, D \,|\, \{C, B\}), \quad I(B, D)$$

Then we can compute the following separators:

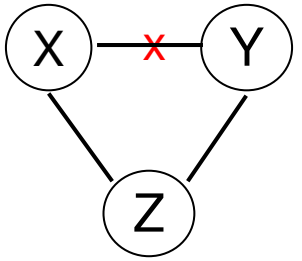
$$S(A, B) := \text{none}$$
$$S(A, C) := \text{none}$$
$$S(A, D) := \{C\}$$
$$S(B, C) := \text{none}$$
$$S(B, D) := \emptyset$$
$$S(C, D) := \text{none}$$

Thus, the skeleton of the Bayesian Network representing I looks like



Computing the V-structure

Lemma 2 (Uncoupled Head-to-head Meeting Criterion)**.** *Let* $G := (V, E)$ *be a DAG,* $X, Y, Z \in V$ *with*



Then it is equivalent:

(i) $X \rightarrow Z \leftarrow Y$ is an uncoupled head-to-head meeting, i.e.,

$$(X, Z), (Y, Z) \in E, (X, Y), (Y, X) \notin E$$

(ii) Z is not contained in any separator of X and Y, i.e.,

$$Z \notin S \quad \forall S \in \mathrm{Sep}(X, Y)$$

(iii) Z is not contained in at least one separator of X and Y, i.e.,

$$Z \notin S \quad \exists S \in \mathrm{Sep}(X, Y)$$

Computing Skeleton and V-structure

```
vstructure(set of variables V, independency relation I) :
S := separators(V, I)
G := (V, E) with E := {{X, Y} | S({X, Y}) = none}
for X, Y, Z ∈ V with X − Z − Y, X ≠ Y do
    if Z ∉ S(X, Y)
        orient X − Z − Y as X → Z ← Y
    fi
od
return G
```

Figure 2: Compute skeleton and v-structure.

```
learn-structure-pc(set of variables V, independency relation I) :
G := vstructure(V, I)
saturate(G)
return G
```

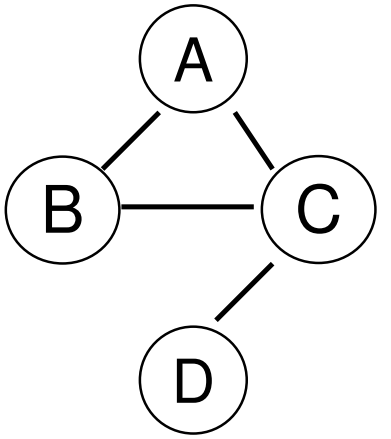
Figure 3: Learn structure of a Bayesian Network (SGS/PC algorithm, [SGS00]).

Example / 2/3 – Computing the V-Structure

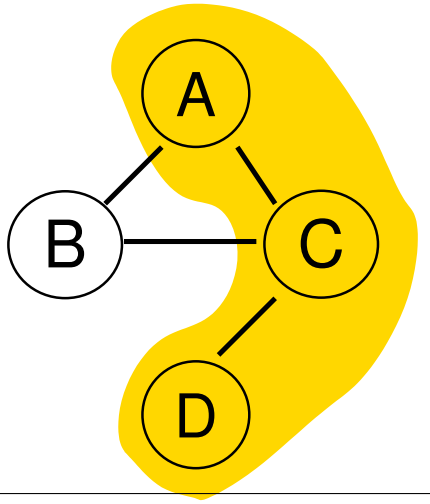
Separators:

$$S(A,B) := \text{none}$$
$$S(A,C) := \text{none}$$
$$S(A,D) := \{C\}$$
$$S(B,C) := \text{none}$$
$$S(B,D) := \emptyset$$
$$S(C,D) := \text{none}$$

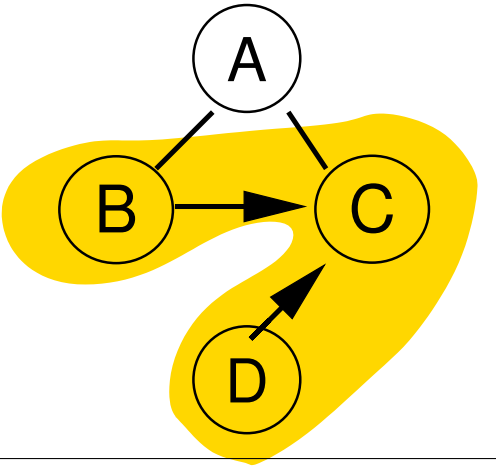
Skeleton:



Checking A–C–D:

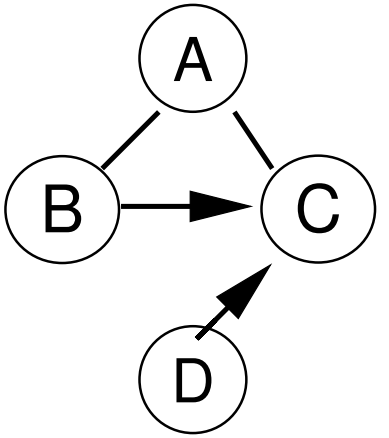


Checking B–C–D:

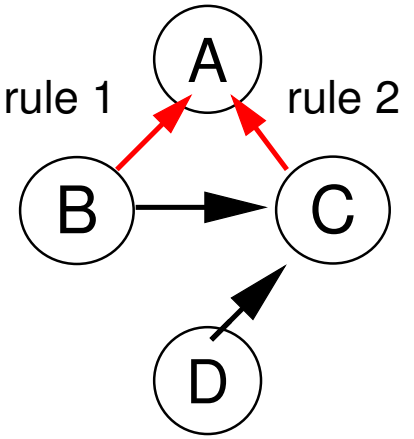


Example / 3/3 – Saturating

Skeleton and v-structure:



Saturating:



Number of Independency Tests

Let there be n variables.

For each of the $\binom{n}{2}$ pairs of variables, there are 2^{n-2} candidates for possible separators.

$$\text{number of } I\text{-tests} = \binom{n}{2} 2^{n-2}$$

Example: $n = 4$:

$$\binom{n}{2} 2^{n-2} = \binom{4}{2} 2^2 = 6 \cdot 4 = 24$$

If we start with small separators and stop once a separator has been found, we still have to check

$$4 \cdot (1 + 2 + 1) + 1 \cdot (1 + 2) + 1 \cdot 1 = 20$$

Number of Independency Tests

Can we reduce the number of tests for a given pair of variable by reusing results for other pairs of variables?

Lemma 3. *Let* $G := (V, E)$ *be a DAG and* $X, Y \in V$ *separated. Then*

$$I(X, Y \mid \mathrm{pa}(X)) \quad \textit{or} \quad I(X, Y \mid \mathrm{pa}(Y))$$

As we do not know directions of edges at the skeleton recovery step, we use the weaker result:

$$I(X, Y \mid \mathrm{fan}(X)) \quad \text{or} \quad I(X, Y \mid \mathrm{fan}(Y))$$

Computing the Skeleton / Separators

```
separators-remove-edges(separator map S, skeleton graph G, independency relation I)
i := 0
while ∃X ∈ V : |fan_G(X)| > i do
    for {X, Y} ∈ E with |fan_G(X)| > i or |fan_G(Y)| > i do
        for T ∈ P^i(fan_G(X) \ {Y}) ∪ P^i(fan_G(Y) \ {X}) do
            if I(X, Y | T)
                S({X, Y}) := T
                E := E \ {{X, Y}}
                break
            fi
        od
    od
    i := i + 1
od
return S
```

```
separators-interlaced(set of variables V, independency relation I) :
Allocate S : P^2(V) → P(V) ∪ {none}
S({X, Y}) := none   ∀{X, Y} ⊆ V
G := (V, E) with E := P^2(V)
separators-remove-edges(S, G, I)
return S
```

Figure 4: Compute a separator for each pair of variables.

Example / Computing the Separators (1/3)

$$I(A, D \,|\, C), \quad I(A, D \,|\, \{C, B\}), \quad I(B, D)$$

$i = 0$:

$A,\ B,\ T = \emptyset$: —

$C,\ T = \emptyset$: —

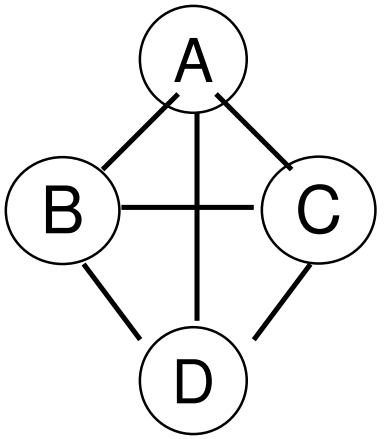
$D,\ T = \emptyset$: —

$B,\ C,\ T = \emptyset$: —

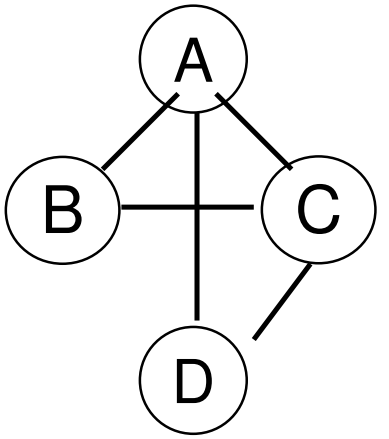
$D,\ T = \emptyset$: $D(\{B, D\}) = \emptyset$

$C,\ D,\ T = \emptyset$: —

initial graph:



after update for B, D:



Example / Computing the Separators (2/3)

$$I(A, D \,|\, C), \quad I(A, D \,|\, \{C, B\}), \quad I(B, D)$$

$i = 1:$

$A,\ B,\ T = \{C\}, \{D\}$: —

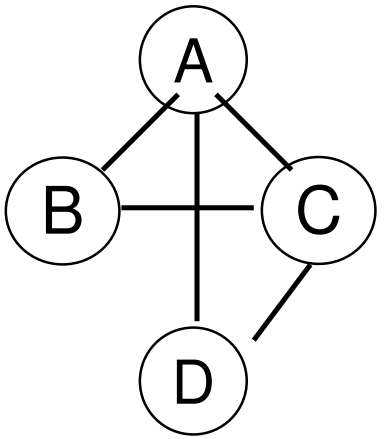
$C,\ T = \{B\}, \{D\}$: —

$D,\ T = \{B\}, \{C\}$: $S(\{A, D\}) = \{C\}$

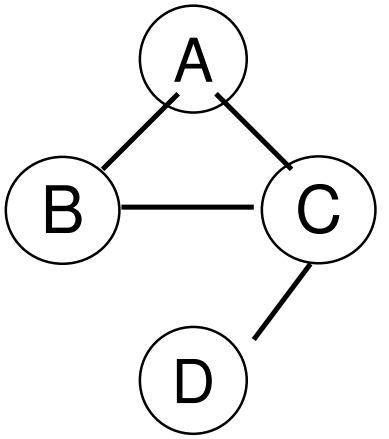
$B,\ C,\ T = \{A\}, \{D\}$: —

$C,\ D,\ T = \{A\}, \{B\}$: —

after update for B, D:



after update for A, D:



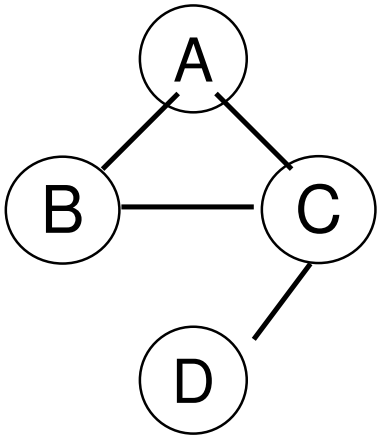
Example / Computing the Separators (3/3)

$$I(A, D \,|\, C), \quad I(A, D \,|\, \{C, B\}), \quad I(B, D)$$

$i = 2:$
$A,\ C,\ T = \{B, D\}$: —
$B,\ C,\ T = \{A, D\}$: —
$C,\ D,\ T = \{A, B\}$: —

total: 19 I-tests.

after update for A, D:



Algorithms – SGS vs. PC

SGS/PC with `separators-basic` is called **SGS algorithm** ([SGS00], 1990).

SGS/PC with `separators-interlaced` is called **PC algorithm** ([SGS00], 1991).

Implementations are available:

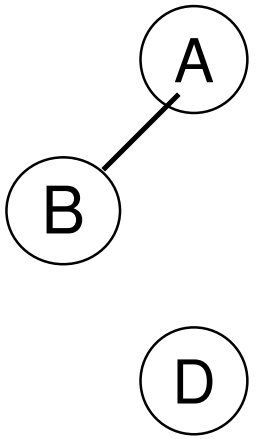
- in Tetrad
  `http://www.phil.cmu.edu/projects/tetrad/`
  (class files & javadocs, no sources)
- in Hugin (commercial).

Another Example

Let I be the following independency structure:

$$I(A, D \mid B), \quad I(B, D)$$

PC computes the following DAG pattern:



But this is not even a representation of I, as it implies

$$I_G(A, D \mid \emptyset)$$

Representation and Faithfulness Tests

PC computes the DAG pattern of an independency relation

if there exists one at all !

Remember: not every independency relation has a faithful DAG representation.

But how do we know if an independency relation has a faithful DAG representation?

Representation and Faithfulness Tests

There is no easy way to decide if a given independency relation has a faithful DAG representation.

So just check ex-post if the DAG pattern we have found is a faithful representation.

Check:

1. compute all $I_G(X, Y \mid \mathcal{Z})$ and check if $I(X, Y \mid \mathcal{Z})$ (representation),
2. check for each $I(X, Y \mid \mathcal{Z})$ if $I_G(X, Y \mid \mathcal{Z})$ (faithfulness).

As there is no easy way to enumerate $I_G(X, Y \mid \mathcal{Z})$ for a DAG pattern G directly, we draw a representative H and then enumerate $I_H(X, Y \mid \mathcal{Z})$ (remember that $I_H = I_G$).

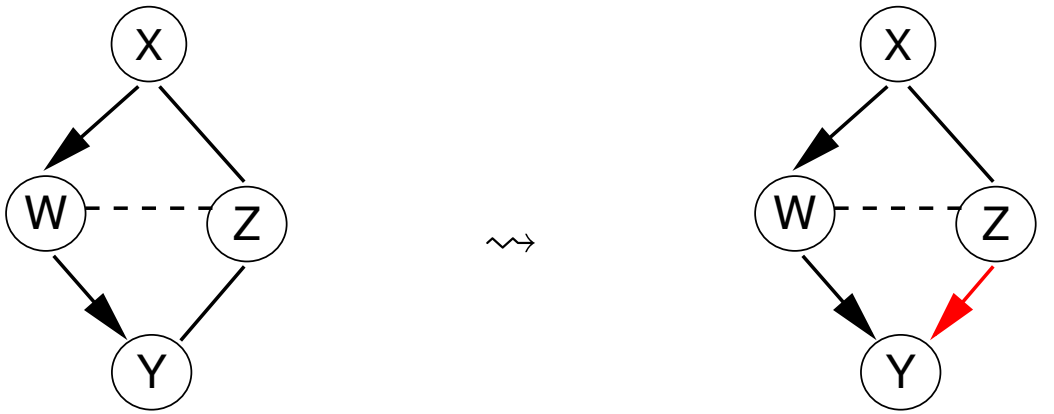
Draw a Representative of a DAG pattern

```
draw-representative(DAG pattern G) :
saturate(G)
while G has unoriented edges do
        draw an edge from G an orient it arbitrarily
        saturate(G)
od
return G
```

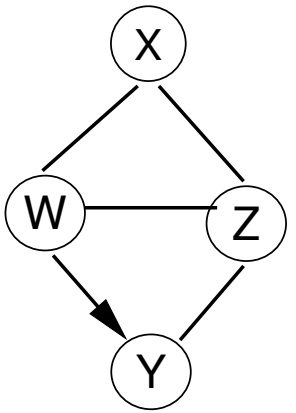
Figure 5: Draw a representative of a DAG pattern.

Draw a Representative of a DAG pattern

Here, rule 4 is necessary in `saturate`.

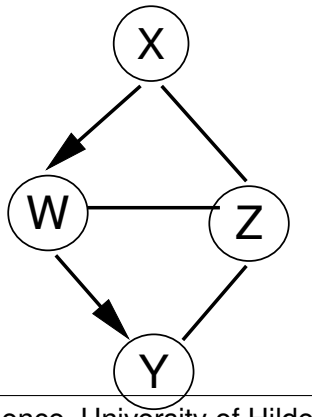


Start with DAG pattern



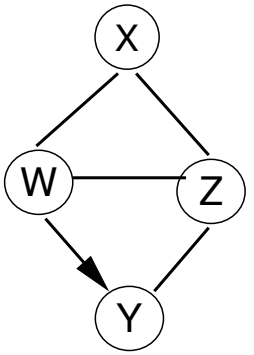
that is already saturated

and choose as next edge X–W and orient it as $X \to W$. Then rule 4 has to be applied to saturate the resulting DAG pattern

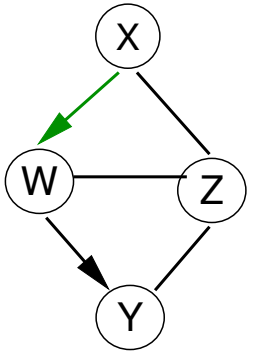


Draw a Representative of a DAG pattern / Example

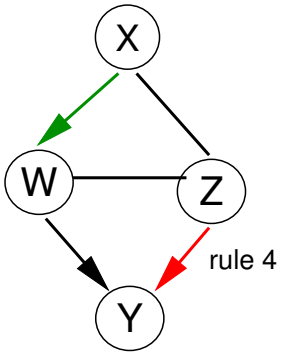
step 1a: saturate.



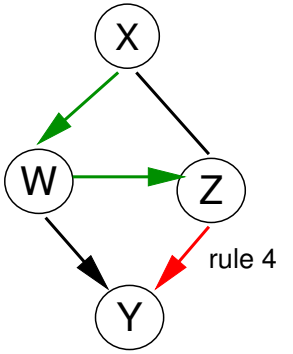
step 1b: orient any unoriented edge:



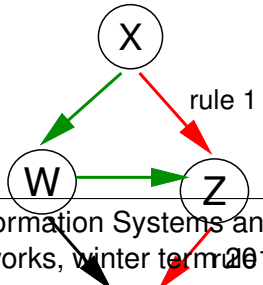
step 2a: saturate.



step 2b: orient any unoriented edge:



step 3a: saturate.



done.

Representation and Faithfulness Tests

If $I = I_p$ the probabilistic independency relation of a JPD p, then for (1) it suffices to check the Markov property, i.e.,

for all X check if $I_p(X, \mathrm{nondesc}(X) \setminus \mathrm{pa}(X) \mid \mathrm{pa}(X))$

It even suffices to check for any topological ordering $\sigma = (X_1, \ldots, X_n)$ if

$$I_p(\sigma(i), \sigma(\{1, \ldots, i-1\}) \setminus \mathrm{pa}(\sigma(i)) \mid \mathrm{pa}(\sigma(i)))$$

A Third Method to Compute Separators

`separators-interlaced` computes separators top-down by

- starting with a complete graph and then
- successively thining the graph.

Therefore, we have to start checking with lots of candidates for possible separators.

A Third Method to Compute Separators

separators-add-edges(separator map S, skeleton graph G, independency relation I) :
$S(\{X,Y\}) := \text{argmin}_{T \subseteq \text{fan}_G(X), T \subseteq \text{fan}_G(Y)}\, g(X,Y,T) \quad \forall \{X,Y\} \in \mathcal{P}^2(V)$
$\{X_0, Y_0\} := \text{argmax}_{\{X,Y\} \in \mathcal{P}^2(V)}\, g(X,Y,S(\{X,Y\}))$
while $\neg I(X_0, Y_0 \mid S(\{X_0,Y_0\}))$ **do**
    $E := E \cup \{\{X_0, Y_0\}\}$
    $S(\{X_0,Y_0\}) := \text{none}$
    $S(\{X,Y\}) := \text{argmin}_{T \subseteq \text{fan}_G(X), T \subseteq \text{fan}_G(Y)}\, g(X,Y,T) \quad \forall \{X,Y\} \in \mathcal{P}^2(V) \setminus E, X \in \{X_0, Y_0\}$
    $\{X_0, Y_0\} := \text{argmax}_{\{X,Y\} \in \mathcal{P}^2(V) \setminus E}\, g(X,Y,S(\{X,Y\}))$
od
return S

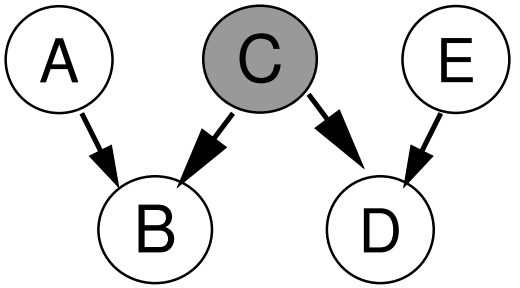
separators-bottumup(set of variables V, independency relation I) :
Allocate $S : \mathcal{P}^2(V) \to \mathcal{P}(V) \cup \{\text{none}\}$
$G := (V, E)$ with $E := \emptyset$
separators-add-edges(S, G, I)
separators-remove-edges(S, G, I)
return S

Figure 6: Compute a separator for each pair of variables [TASB03].

Embedded Faithfulness

Let I be the following independency structure:

$$I(A, D), \quad I(A, E), \quad I(A, E \mid B), \quad I(A, E \mid D), \quad I(B, E)$$



Assume C to be hidden.

d-separations between variables A,B,D, and E:

$$I(A, D), \quad I(A, E), \quad I(A, E \mid B), \quad I(A, E \mid D), \quad I(B, E)$$

Embedded Faithfulness

Definition 2. Let I be an independency relation on the variables V and G be a DAG with vertices $W \supseteq V$.

I **is embedded in** G, if all independency statements entailed by G between variables from V hold in I:

$$I_G(\mathcal{X}, \mathcal{Y} \mid \mathcal{Z}) \Rightarrow I(\mathcal{X}, \mathcal{Y} \mid \mathcal{Z}) \quad \forall \mathcal{X}, \mathcal{Y}, \mathcal{Z} \subseteq V$$

I **is embedded faithfully in** G, if the independency statements entailed by G are exactly I:

$$I_G(\mathcal{X}, \mathcal{Y} \mid \mathcal{Z}) \Leftrightarrow I(\mathcal{X}, \mathcal{Y} \mid \mathcal{Z}) \quad \forall \mathcal{X}, \mathcal{Y}, \mathcal{Z} \subseteq V$$

Many independency relations w./o. faithful DAG representation can be embedded faithfully.

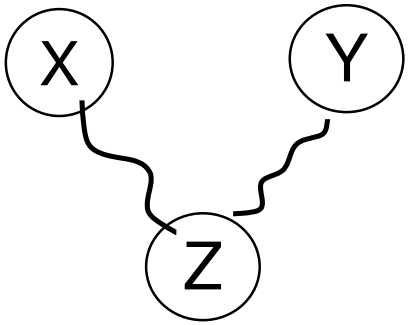
But not every independency relation can be embedded faithfully.

Embedded Faithfulness / Example

Let I be the independency relation [Nea03, ex. 2.13, p. 103]

$$I(X, Y), \quad I(X, Y \mid Z)$$

Assume, I can be embedded faithfully in a DAG G.



- As $\neg I(X, Z)$, there must be chain $X \sim Z$ w./o. head-to-head meetings.
- As $\neg I(Z, Y)$, there must be chain $Y \sim Z$ w./o. head-to-head meetings.

Now, concatenate both chains $X \sim Z \sim Y$:

- eihter $X \sim\!\rightarrow Z \leftarrow\!\sim Y$, and then the chain is not blocked by Z, i.e., not $I(X, Y \mid Z)$,
- or not $X \sim\!\rightarrow Z \leftarrow\!\sim Y$, and then the chain is not blocked by $\emptyset$, i.e., not $I(X, Y)$.

There are variants of the PC algorithm for finding faithful embeddings of a given independency relation:

- Causal Inference (CI) and
- Fast Causal Inference Algorithms (FCI; [SGS00])

See also [Nea03, ch. 10.2].

Outline of Part 1 : Learning Bayesian Networks

section	*key concepts*
I. Probabilistic Independence and Separation in Graphs	Prob. independence, separation in graphs, Bayesian Network
II. Learning Parameters	Max. likelihood estimation, max. a posterior estimation
III. Learning Parameters with Missing Values	EM algo.
IV. Learning Structure by Constrained-based Learning	Markov equivalence, graph patterns, PC algo.
V. Learning Structure by Local Search	Bayesian Information Criterion (BIC), K2 algo., GES algo.

References

[Nea03] Richard E. Neapolitan. *Learning Bayesian Networks*. Prentice Hall, 2003.

[SGS00] Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. MIT Press, 2 edition, 2000.

[TASB03] I. Tsamardinos, C. F. Aliferis, A. Statnikov, and L. E. Brown. Scaling-up bayesian network learning to thousands of variables using local learning technique. Technical report, DSL TR-03-02, March 12, 2003, Vanderbilt University, Nashville, TN, USA, 2003.